

Computing shortest paths in 2D and 3D memristive networks

Zhanyou Ye,^{1,*} Shi Hong Marcus Wu,^{2,†} and Themistoklis Prodromakis^{1,3,‡}

¹*Centre for Bio-Inspired Technology, Department of Electrical and Electronic Engineering,
Imperial College London, London SW7 2AZ, United Kingdom*

²*Systems Engineering Advancement Research Initiative (SEARI), Massachusetts Institute of Technology,
77 Massachusetts Avenue, E38-576 Cambridge, MA 02139-4307*

³*School of Electronics and Computer Science, University of Southampton, Southampton SO17 1BJ, United Kingdom*

(Dated: March 20, 2013)

Global optimisation problems in networks often require shortest path length computations to determine the most efficient route. The simplest and most common problem with a shortest path solution is perhaps that of a traditional labyrinth or maze with a single entrance and exit. Many techniques and algorithms have been derived to solve mazes, which often tend to be computationally demanding, especially as the size of maze and number of paths increase. In addition, they are not suitable for performing multiple shortest path computations in mazes with multiple entrance and exit points. Mazes have been proposed to be solved using memristive networks and in this paper we extend the idea to show how networks of memristive elements can be utilised to solve multiple shortest paths in a single network. We also show simulations using memristive circuit elements that demonstrate shortest path computations in both 2D and 3D networks, which could have potential applications in various fields.

I. INTRODUCTION

Many combinatorial optimisation problems in graph theory [1], such as the Travelling Salesman Problem, involve deriving the shortest path within networks [2]. Applications of such computations include optimising routing protocols [3], transportation models [4] and recurrent neural networks [5]. Perhaps the simplest shortest path problem is a traditional maze, where one has to determine the path to the exit of a labyrinth whilst only given the entrance point. However, when there are a larger number of pathways in a maze, this increases the number of solutions. Out of these possible pathways, finding the shortest or least-cost one may not necessarily be straightforward. Many mathematical algorithms have been proposed to solve mazes, such as random mouse or mathematical search algorithms [6, 7]. Such algorithms derive solutions in a sequential fashion, thus solution times can increase exponentially in complex networks.

There are also many innovative methods prescribed to solve mazes using biological and chemical systems, such as amoeboid organisms [8, 9], chemotaxis [10] and chemotactic droplets [11]. However, such methods also suffer from increased time complexity when maze sizes increase. The problem is further exacerbated with the introduction of multiple users to a network, such as a traffic optimisation problem where multiple cars would like to find the shortest travelling path in order to avoid congestion in the network. In this paper, we propose using networks of memristive elements to perform multiple shortest path computations in a given network.

The memristor, short for memory resistor, is a passive two-terminal circuit element capable of altering its resistance based on the input and remember its past dynamics [12]. After the device was postulated by L.Chua, a generalised concept of the memristor was further proposed by Chua and Kang [13], defined as

$$v = R(x)i \quad (1)$$

$$\frac{dx}{dt} = f(x, i) \quad (2)$$

where v represents the voltage, i represents the current and $R(x)$ denotes the instantaneous resistance of the device that changes based on its internal state variable, x [14]. Memristance signatures are also observed in various dissipative systems that support discharge phenomena, such as discharge lamps and biological ion channels [15, 16]. Since its implementation by Strukov et.al [17], the solid-state memristor has been proposed to be of use in various applications such as memory storage [18] and neuromorphic implementations [19, 20].

Memristor networks - several memristors connected in the form of an array- have been postulated to be able to perform complex cortical computing functions [20]. Pershin and Di Ventra have also demonstrated that abstract mazes can be solved in a parallel fashion using memristive networks, a termed coined as *analog parallelism*. They have also shown that all solutions in the maze can be determined and the results are separated in order of path length [21].

In the rest of the paper, we exploit the plasticity of 2D and 3D memristive networks for extrapolating various shortest path solutions via simulations in PSPICE. Our study initiates by deciphering the fundamentals of the network to derive shortest path solutions to a given maze in 2D. We then expand this concept to exhibit how a

* zhanyou.ye09@imperial.ac.uk

† marcuswu@mit.edu

‡ t.prodromakis@soton.ac.uk

memristive grid can be used to perform multiple shortest path computations in a network involving several users; the example here is London's Tube Network. Lastly, we show how multiple shortest path problems can be solved using 3D memristive networks.

II. METHODOLOGY

The maze or network must first be mapped onto a regular memristive grid. The representative memristive grid is then implemented in MATLAB and the corresponding circuit simulations are performed using PSPICE. Entrance/Exit or Start/End nodes for the circuit simulation are represented by a 1V DC Voltage source and Ground respectively.

A. MEMRISTIVE COMPONENTS

In 1971, the memristor was predicted theoretically by L. Chua in his seminal paper [12] but it remained a theoretical abstraction until researchers at Hewlett Packard (HP) Laboratories discovered similar properties while fabricating crossbar-type nano-devices in 2008 [17, 22]. The memristor was postulated based on a mathematical relationship between charge q and magnetic flux φ : $d\varphi = Mdq$, where M denotes the memristance, which has the same units as resistance (Ω) and is defined as the resistance across the memristor. By taking the time integrals of q and φ , the non-linear relationship between voltage and current across the memristor is established:

$$v(t) = M(q) * i(t) \quad (3)$$

In Eq (3), $v(t)$ is the applied bias, $i(t)$ is the current flowing through the memristor and $M(q)$ is the charge-dependent memristance. The simplest abstraction of the memristor is that of a time-dependent resistor [23]:

$$M(t) = \frac{W(t)}{D} * R_{ON} + (1 - \frac{W(t)}{D}) * R_{OFF} \quad (4)$$

B. MEMRISTIVE FUSE

The conductance modulation of a single memristor depends on the polarity of the charge flowing through it. However, when devices are used in memristive networks for shortest path computations, polarity dependence is not desirable since the direction of current flow cannot always be determined.

It was previously proposed that by connecting two memristors with opposing polarities [24], the non-linear relationships between time integrals of voltage and current can be preserved without any polarity dependence. This new combination of devices is termed the memristive fuse [25], shown in Fig. 1, and is used as the primary memristive device in all the networks in the following

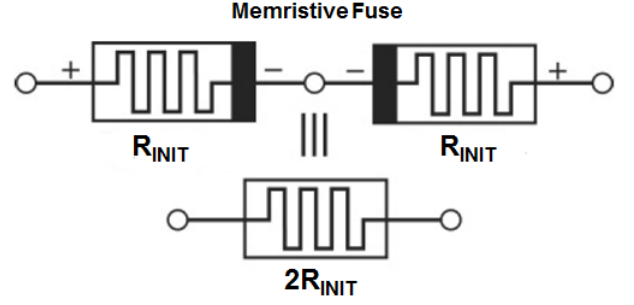


FIG. 1. Two memristors connected with opposite polarities to form a Memristive Fuse [25].

simulations. Since the fuse is made of two ideal memristors connected in series, the total initial resistance of the overall device is twice the initial resistance of a single memristor ($2R_{INIT}$).

C. SPICE SIMULATIONS

All the memristive networks are simulated using Bi-olek's Memristor SPICE Model [26] with Prodromakis' non-linear kinetics dopant model [27]. Details about the models can be found in the corresponding references. Throughout this study, the memristor parameters within the SPICE model were defined as follows: Initial Width $W_0 = 5 \times 10^{-9}\text{m}$, Active-Core Thickness $D = 10 \times 10^{-9}\text{m}$, ON Resistance $R_{ON} = 100\Omega$, OFF Resistance $R_{OFF} = 16\text{k}\Omega$, Net Resistance at $t = 0$ $R_{INIT} = 1000\Omega$, Mobility $\mu = 1 \times 10^{-14}\text{m}^2\text{s}^{-1}\text{V}^{-1}$.

III. SHORTEST PATH SOLUTION OF MAZES USING 2D MEMRISTIVE NETWORKS

The first simulation shows the varying conductance paths in a memristive network, which correspond to the various solutions to a simple shortest path computation. A simple memristive network is first constructed from a combination of memristive fuses and resistors. Throughout this work we refer to the points where devices are connected together as nodes, while we refer to a branch in the case it comprises one or more devices between two nodes. In Fig. 2, a simple maze is illustrated using a 4×4 memristive network. The paths of the maze are simulated using 12 memristive fuses (labelled M1 - M12) and $2\text{ M}\Omega$ resistors are used to represent the blocked conductance paths. A 1V DC Voltage Source and Ground are placed at the nodes corresponding to the entrance and exit of the maze respectively.

The memristive network is simulated for 35s and the results are shown in Fig. 3 and Fig. 4. For each device, the change in memristance ΔM is determined by taking the difference between the resistance across each branch

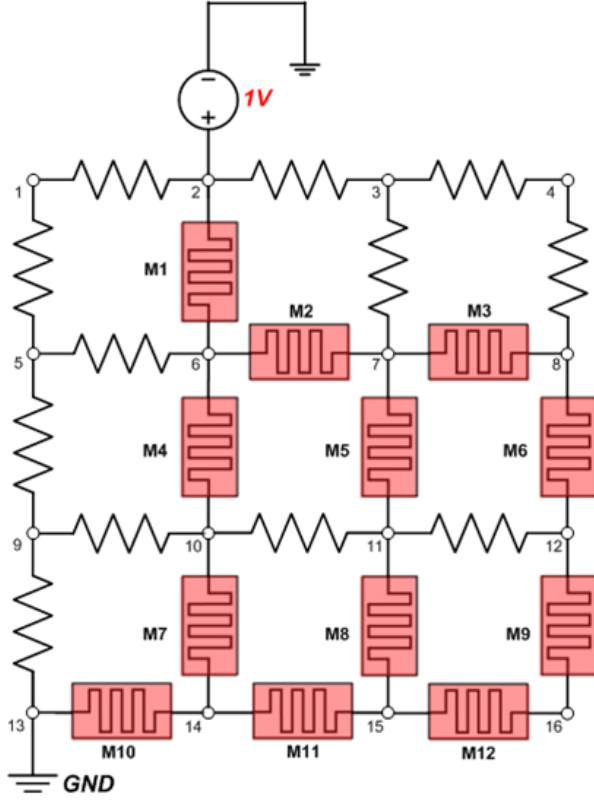


FIG. 2. (Colour Online) 4 x 4 Memristive Network consisting of Memristive Fuses and Resistors.

and the initial resistance ($2R_{INIT}$), which is calculated via:

$$\Delta M_k = \left| \frac{V_X - V_Y}{I_{XY}} \right| - 2R_{INIT} \quad (5)$$

x and y are the nodes connecting each branch and k is the device number. Fig. 3 illustrates the transient response of ΔM for devices M4, M5 and M6 against time. This change shows how the devices in different paths respond to the input voltage due to the variance in current amplitudes flowing through them. Fig. 4 shows the temporal evolution of memristance of all 12 devices in the network for time instances 1s, 5s, 10s and 30s. For better visualisation of the change in memristance, ΔM for each device was translated to a linear colour scale of 0 - 64, where 0 corresponds to zero ΔM and 64 represents the maximum ΔM observed throughout the duration of the simulation.

We first analyse the memristance change of the devices between three branches: Branch 1 (nodes 6 and 14), Branch 2 (nodes 7 and 15) and Branch 3 (nodes 8 and 16). Kirchoffs Current Law states:

$$I_{in} = \sum_{i=1}^n I_i \quad (6)$$

n refers to the number of branches at the particular node. Applying the formula at nodes 14 and 15, the following

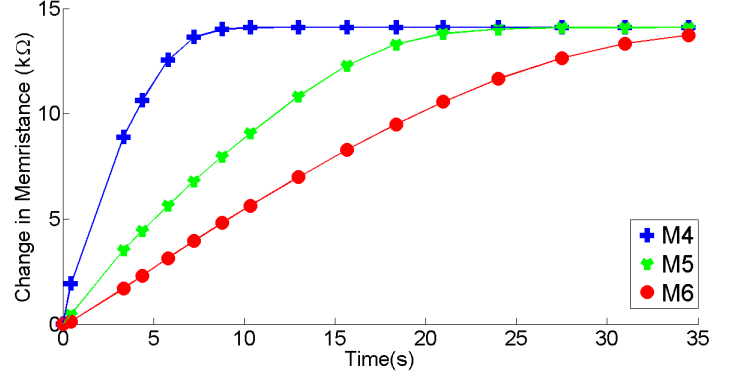


FIG. 3. (Colour Online) Plot of Memristance Change (Devices M4, M5 and M6) for time period 0 - 35s.

relationship regarding the overall current flow across all three branches can be deduced: $I_{B3} < I_{B2} < I_{B1}$.

A larger current flow across a memristor will result in higher rate of change of memristance of the device. Assuming that very little current flows through the $2 M\Omega$ resistors in the grid network, this implies that the ΔM of the devices in all three branches after a short time period will have a similar relationship to that of the total initial current flow across the branches: refer to the change in memristance across devices M4, M5 and M6, shown in Fig. 3 respectively. An increase in the memristance across the devices in Branch 1 will in turn channel more current to Branches 2 and 3. After a stipulated simulation time, all the memristive devices in the network will reach the high resistive state (R_{OFF}).

It can be observed that the shortest path in a maze will exhibit a larger change in memristance over a single period. For the simple maze in Fig. 2, Branch 1 is clearly the shortest path, and the corresponding length of the other two paths (Branch 2 followed by 3) can be identified by comparing ΔM of all devices. We also note the limit of the simulation where all devices reach the high resistive state (R_{OFF}) and the paths are no longer distinguishable by measuring ΔM . Based on the argument that discharge-phenomena support memristive signatures [15], we have reviewed numerous reports on unconventional computation via discharge mechanisms; the most prominent one being [28]. The maze, shown in Fig. 5a, is reproduced by Reyes et.al [28], where the solution is determined using an analog computation method via glow discharge in microfluidic chips (Fig. 5b). The maze is first mapped onto a 15 x 15 memristive grid and the red and blue lines of the grid overlapped onto the maze represent memristive fuses and $2 M\Omega$ resistors respectively, as shown in Fig. 5c. A 1V DC Voltage Source and Ground are placed at the entrance and exit nodes respectively. Spatiotemporal representation of the change in memristance ΔM of the memristive devices are shown in Fig. 5d, Fig. 5e and Fig. 5f for the times 2s, 6s and 10s respectively, and the shortest path solution is shown to be

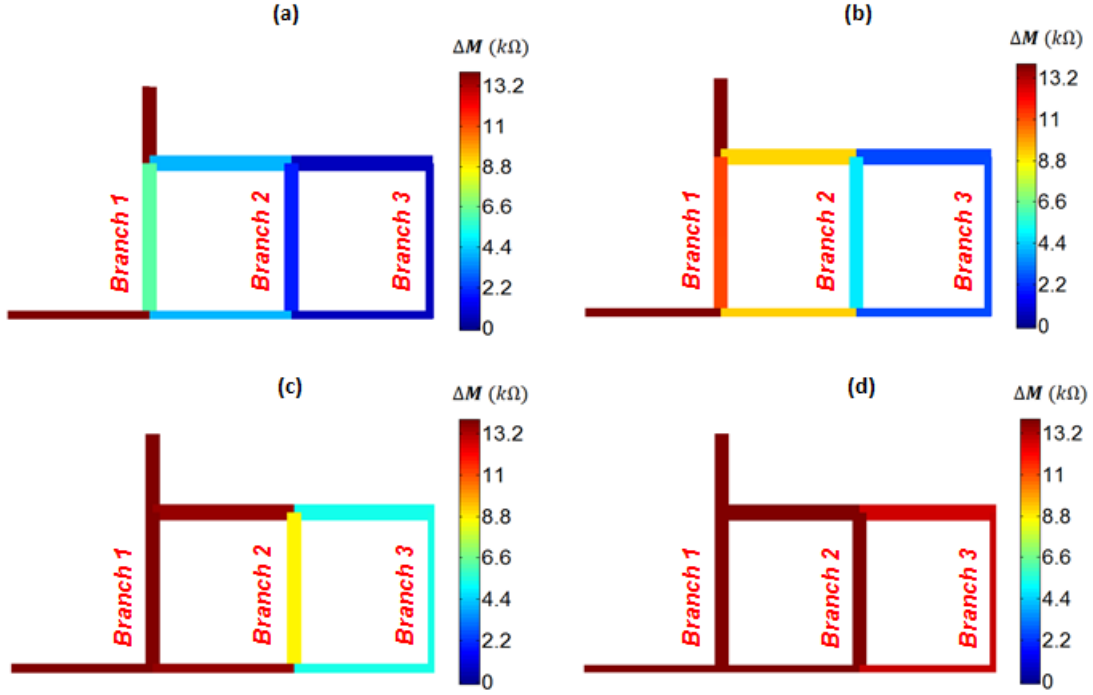


FIG. 4. (Colour Online) Spatiotemporal plot depicting ΔM of all Memristive Elements at varying times: (a) 1s, (b) 5s, (c) 10s, (d) 30s. The colour bar on the right show the corresponding ΔM values.

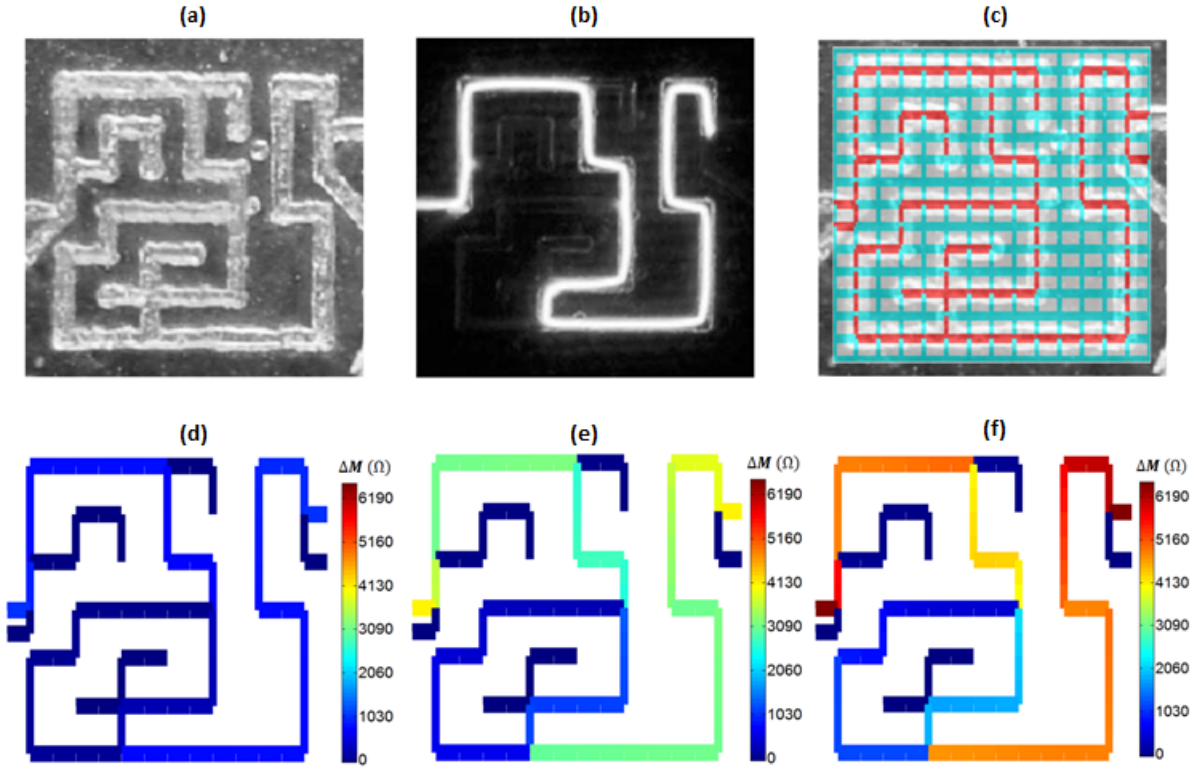


FIG. 5. (Colour Online) Maze (a) and Solution (b) shown by Reyes et al [28]. (c) Mapping of Maze to a 15 x 15 Memristive Grid. Solutions to maze shown by simulations after 2s (d), 6s (e) and 10s (f).

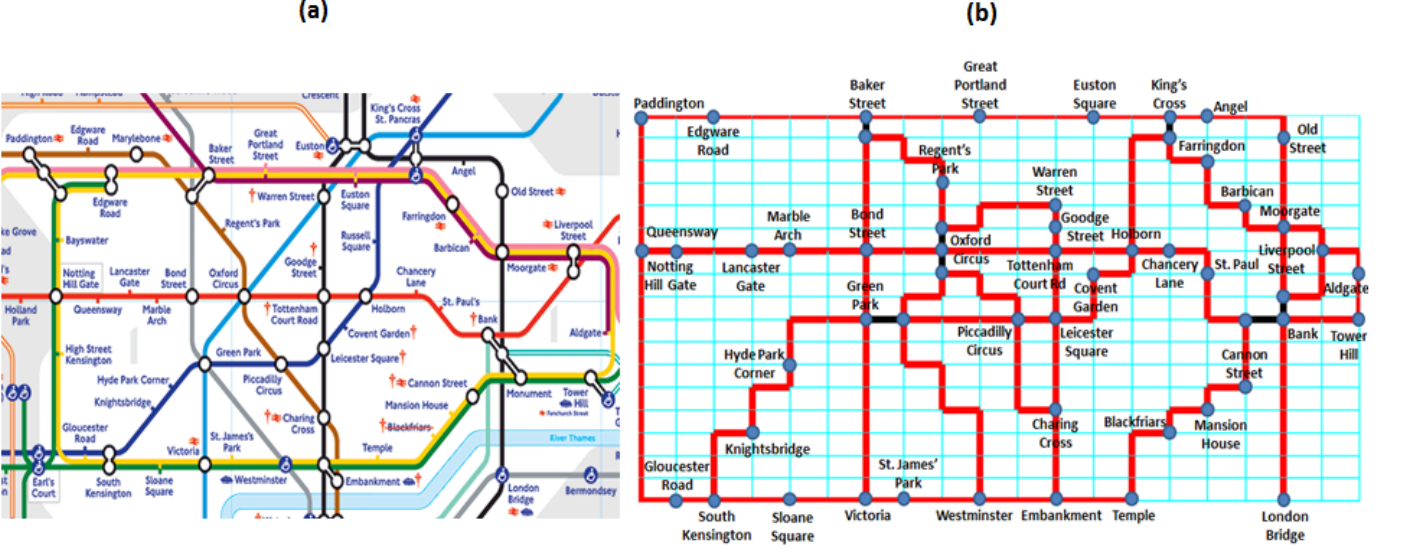


FIG. 6. (Colour Online) Zone 1 of London's Tube Network (a) is mapped onto an 18 x 20 Memristive Grid (b).

identical when compared with the solution derived using microfluidic chips.

This case verifies that the solution to a maze can indeed be determined by mapping it to a memristive grid and placing the source and ground at the entrance/exit nodes. At the same time, this example proves the concomitantly argument presented in [15]: discharge phenomena manifest memristive signatures. By exploiting the analog computations facilitated by Kirchoff's Current Law and that current follows the shortest path to ground, the shortest conductance path will exhibit the largest ΔM . In addition, the altered devices will stay at their given resistive states even after the source and ground nodes have been removed.

In this example, the memristive network converged to a possible solution to the maze after a simulation time of approximately 6 - 10s. Nonetheless, this approach is clearly amenable to the use of larger biasing potentials that will in turn speed up the solution. It is interesting to compare this to other mathematical search algorithms performed by a micro-mouse robot; a robot searching for the shortest path in a 16 x 16 unit square maze using either Dijkstra's [29] or Flood-Fill algorithms typically requires 100s of seconds to accomplish similar tasks [7, 30]. Although actual memristive hardware implementations may yield different solution times from software simulation results, this comparison gives us a scale of the improvements in time complexity by utilising such memristive networks.

IV. MULTIPLE SHORTEST PATH COMPUTATIONS USING 2D MEMRISTIVE NETWORKS

So far, we have seen the computation of shortest paths for mazes with fixed entrance and exit points. In this section, we further elaborate on the possibility of concurrently solving multiple shortest paths within a same network via an example of travellers determining the shortest path on London's Tube Network. Zone 1 of London's Tube Map, shown in Fig. 6a, is first mapped onto an 18 x 20 memristive grid, as illustrated in Fig. 6b. Similarly, the red and blue lines on the grid represent memristive fuses and 2 M Ω resistors respectively. The mapping is an approximation of the actual distances and time taken between the tube stations and solely for the demonstration of shortest path computations between stations.

One of the limitations of performing simulations using 2D memristive grids is that each centre node and corner node can accommodate a maximum of four and two paths passing through them respectively. While investigating the Tube Network application, this limitation in paths per node is insufficient for representing some stations such as Green Park, which has six lines going in and out of the station. Hence, in order to increase the number of possible paths through each node without increasing the dimensional space of the network, a 1 Ω resistor is hereby used to link two neighbouring nodes to increase the node size. These extensions are shown as black lines in Fig. 6b and they signify that the two nodes are now effectively the same station. In this scenario, the voltage drop across the resistor is assumed to be negligible since the corresponding resistance is three orders of magnitude smaller than the initial resistance R_{INIT} of the memristors used in the circuit. All starting and des-

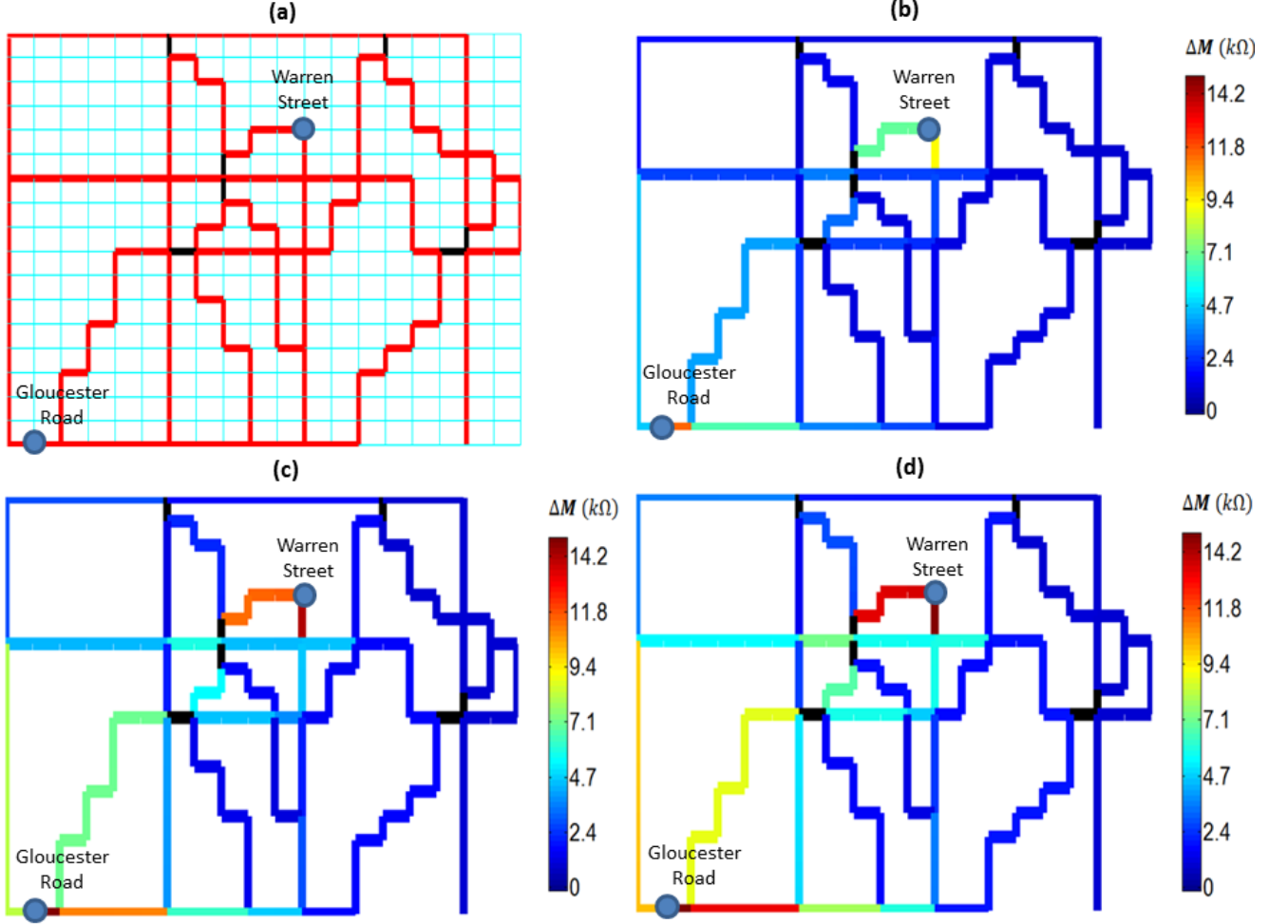


FIG. 7. (Colour Online) Traveller A heading from Gloucester Road to Warren Street Station. The respective positions of nodes are shown on the network in (a). Solution to the network shown by simulations after 3s (b), 7s (c) and 10s (d).

termination nodes in the memristive network are simulated using 1V DC Voltage Sources and Ground.

We first show the shortest path computation in the Tube network for a single traveller wishing to get from Gloucester Road Station to Warren Street Station. Fig. 7 shows the corresponding results of the network for the times 3s, 7s and 10s and the shortest path is accurately determined by observing the spatiotemporal plot of ΔM for all the memristive devices in the network. Moreover, we demonstrate how the memristive network computes shortest paths for three travellers, namely Travellers A, B and C in the tube network concurrently. In the first scenario, all three travellers wish to get to the same destination, Holborn from their respective starting stations: A - Paddington, B - Gloucester Road and C - London Bridge. At the circuit level implementation of the memristive network, this translates to three 1V DC sources at the starting nodes and a single ground placed at the node representing Holborn station. The shortest paths of the three travellers will be termed P_A , P_B and P_C re-

spectively and are shown in the memristive network in Fig. 8.

By comparing the relative ΔM on the spatiotemporal plot shown in Fig. 8, we note that there are two possible shortest paths solutions, P_{A1} and P_{A2} . The number of memristive elements (N) for the two solutions are $N_{A1} = 19$ and $N_{A2} = 20$, as shown in Fig. 8. As the memristive network size increases, the average N increases as well. If the difference in path lengths, $(N_1 - N_2) \ll N_x$ (where $x = 1$ or 2), it will be increasingly difficult to distinguish between two shortest paths using ΔM of the devices as $\langle \Delta M_{P1} \rangle \approx \langle \Delta M_{P2} \rangle$, where $\langle \Delta M_{P1} \rangle$ and $\langle \Delta M_{P2} \rangle$ are the average ΔM of the memristive devices in paths 1 and 2 respectively. In the second scenario however, shown in Fig. 9, Travellers A, B and C all have different start and end stations: Traveller A wishes to get from Gloucester Road to Paddington, Traveller B from Hyde Park Corner to Holborn and Traveller C from London Bridge to Old Street. It is noted that all computed paths are unique solutions; there are no overlapping paths between the

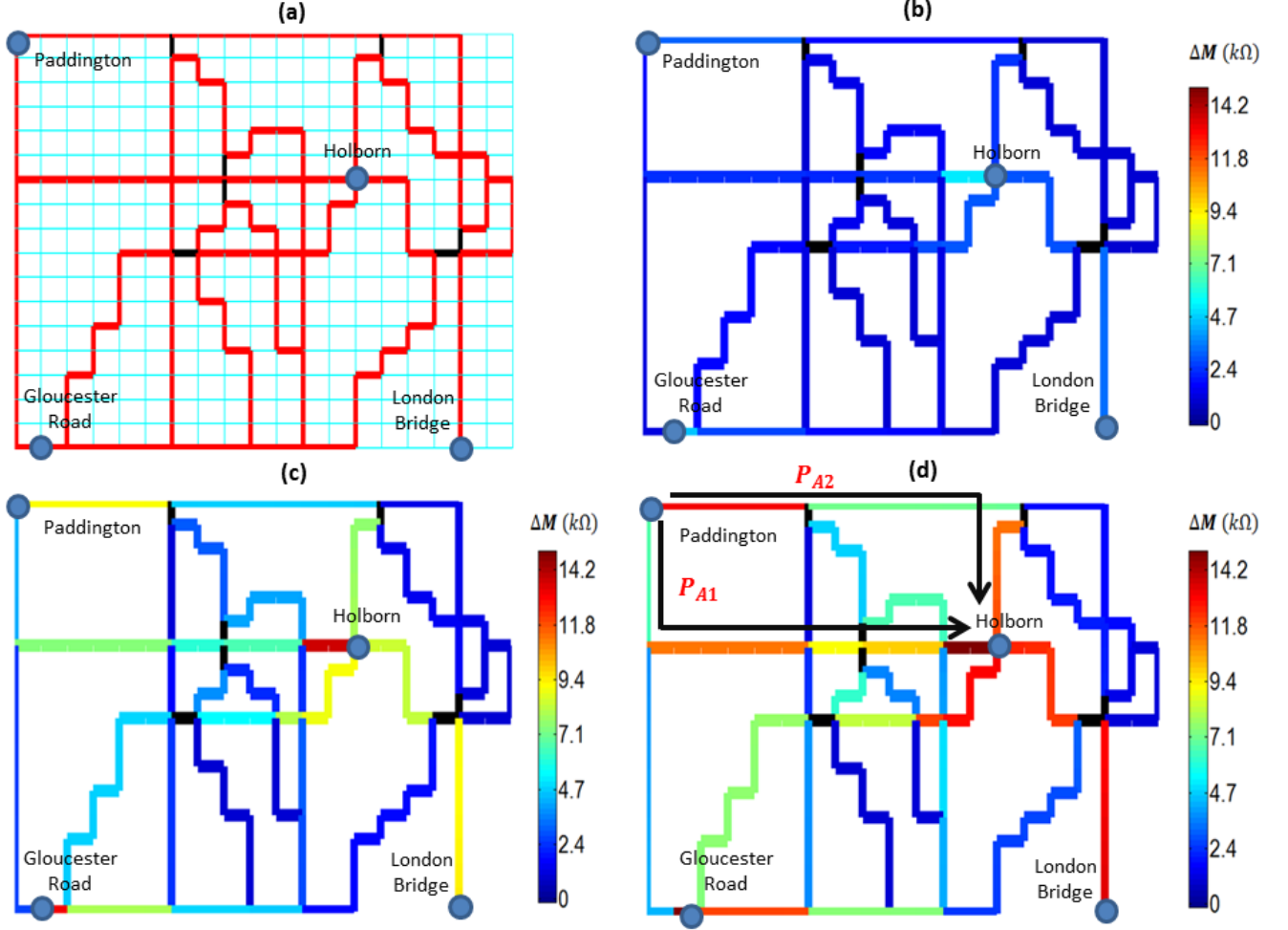


FIG. 8. (Colour Online) Travellers A, B and C heading from Paddington, Gloucester Road and London Bridge Stations to Holborn Station. The respective positions of nodes are shown on the network in (a). Solution to the network shown by simulations after 1s (b), 5s (c) and 10s (d). Black arrows shown in (d) indicate the two paths for Traveller A (Paddington to Holborn).

travellers.

Due to the use of voltage sources at the starting node of the route, the shortest path computed for one traveller will not pass through the starting point of another. This is shown in another example, when Travellers A and B travel from Gloucester Road and Notting Hill Gate to Paddington respectively. As seen from the shortest path computations presented in Fig. 10, the path computed by the memristive network for Traveller A does not pass through Notting Hill Gate station although that path has a lower N value. In the circuit implementation, both station nodes are at high voltage potential, hence resulting in a negligible amount of current flow between them. Even after 10s, the measured ΔM of the devices between the two nodes is approximately only 2Ω . The shortest path for Traveller A will essentially be the next alternative path, as marked out by the red arrow in Fig. 10.

This series of cases exhibit that multiple shortest path

computations can be performed based on the overall change in memristance due to the current flows in a single 2D memristive network. This has been demonstrated using London's Tube Network, where the shortest paths of three travellers are determined concurrently using a single network. If other known shortest path algorithms such as Dijkstra's [29] were used in this example, routes for the three travellers will have to be determined independently, which increases the time complexity of computation by an order of the number of travellers there are in a network. By computing the shortest paths in a parallel manner by solving a series of Kirchoff's Current Law equations, the memristive grid is able to compute all shortest paths in a single step. In addition, all the solutions are shown over a fixed time period regardless of the number of travellers in the network.

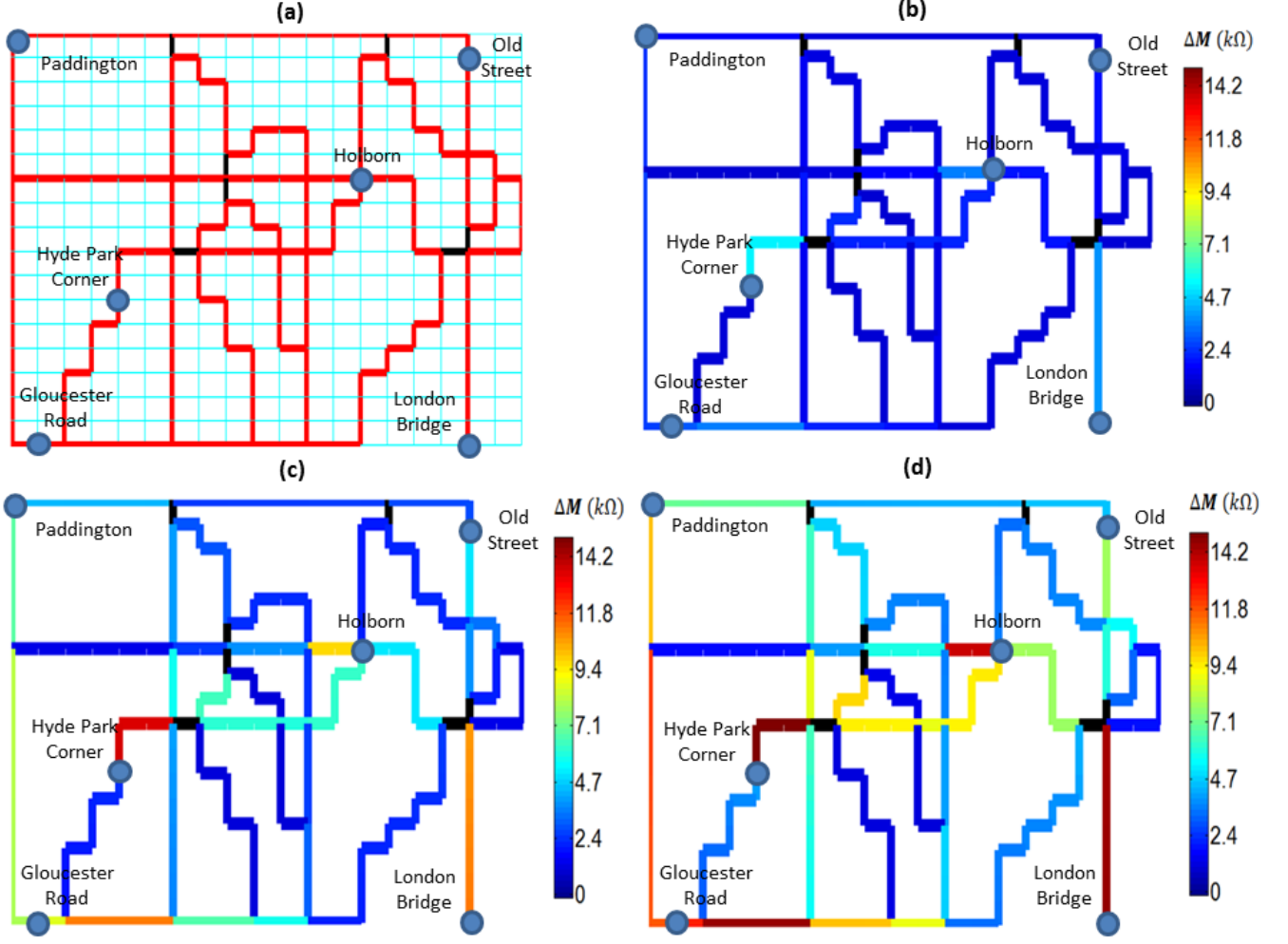


FIG. 9. (Colour Online) Travellers A, B and C on three routes: Gloucester Road to Paddington, Hyde Park Corner to Holborn and London Bridge to Warren Street. The respective positions of nodes are shown on the network in (a). Solution to the network shown by simulations after 1s (b), 5s (c) and 10s (d).

V. SHORTEST PATH COMPUTATIONS USING 3D MEMRISTIVE NETWORKS

The limitations using 2D networks are fewer input and output paths per node, in addition to the relatively low spatial resolution that can be achieved. For example, if all the lines in London's Tube Network (Zones 1-5) were to be mapped onto a single memristive network, it will be more accurately performed in 3D, where an additional layer can accommodate overlapping lines in the Tube network. In this section, we describe the computation of shortest paths by employing 3D grids, using a simple maze constructed in a $4 \times 4 \times 3$ 3D memristive network with two entrances and a single exit. The paths for the maze, represented using memristive fuses in the corresponding circuit are shown in Fig. 11a as light blue lines, while all static resistive elements are represented by thin black lines. The corresponding circuit is exploited in a similar manner to the pre-discussed scenarios. The two

shortest path solutions of the maze, shown in Fig. 11b, Fig. 11c and Fig. 11d, are clearly depicted by monitoring the ΔM of all memristive devices.

A 3D network can also be viewed as several 2D arrays stacked onto each other, with the addition of linking elements between the layers. We compare the time complexity of solving a 3D maze if any random mouse algorithm is used [7]. Assuming that the number of vertices and paths in each layer remain the same, the total time complexity for the random mouse method will increase by an order of the number of 2D arrays, including the number of interconnecting paths.

This computation method via 3D memristive networks has been proven to be simple to execute and does not require long computation times. Current shortest path algorithms such as Dijkstra's [29] and Floyd-Warshall's [31] have time complexities of $O(V^2)$ and $O(V^3)$ respectively where V is the number of vertices (nodes) [32]. In comparison to the employed memristive network, the

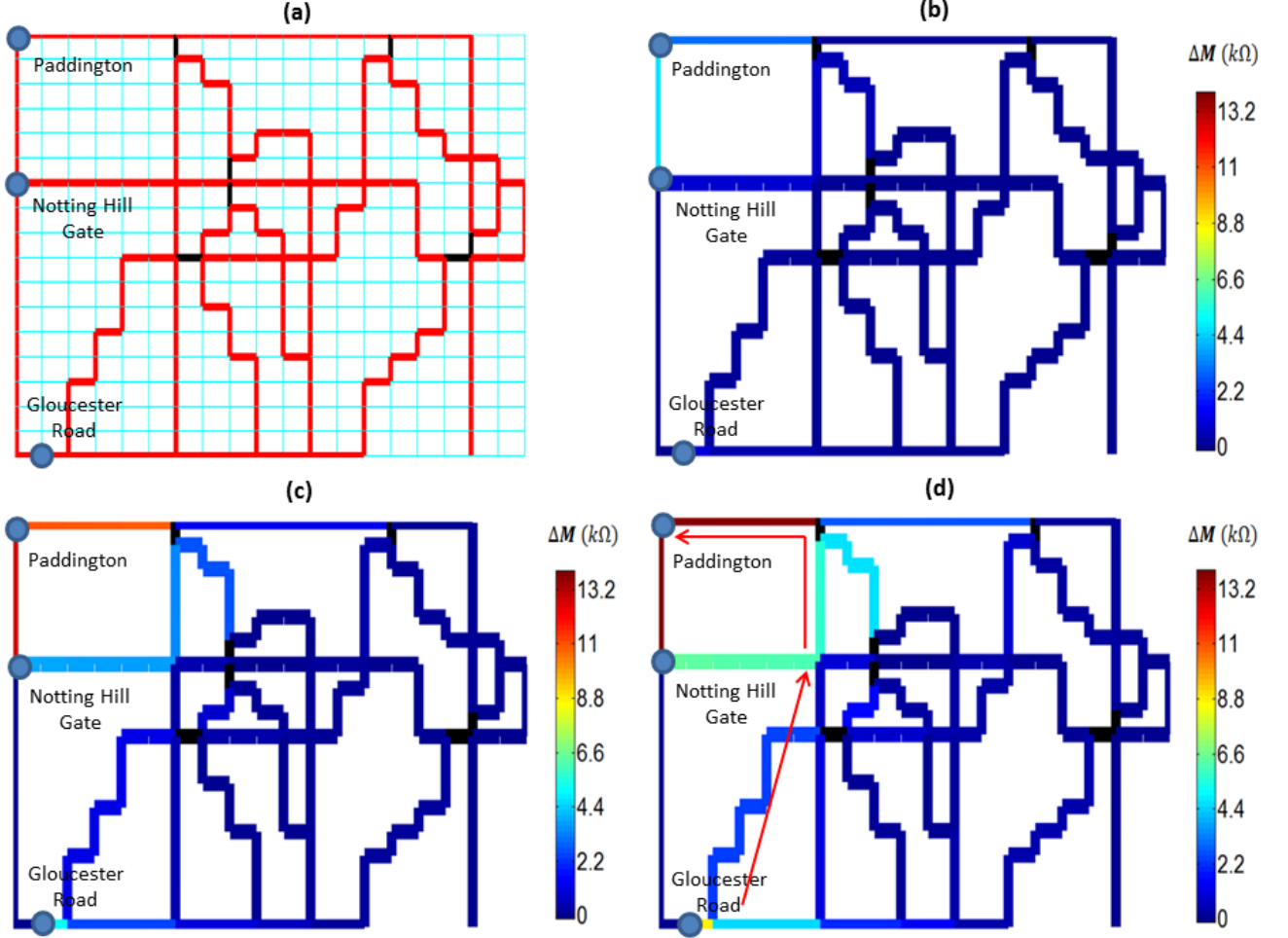


FIG. 10. (Colour Online) Travellers A and B heading from Gloucester Road and Notting Hill Gate to Paddington Station. The respective positions of nodes are shown on the network in (a). Solution to the network shown by simulations after 1s (b), 5s (c) and 10s (d). The red arrow shown in (d) indicates the alternate path for Traveller A.

best theoretical estimate for a linear system is the Coppersmith Winograd algorithm [21, 33] which is described as $O(n^{2.376})$ where n is the number of edges in a network. However, it is noted that a memristive network implemented in hardware only has a single overall computation step in order to determine the shortest paths in the network [21]. This makes it more efficient than the algorithms listed above, before even considering multiple computations in a single network.

VI. CONCLUSION

Paths of an existing network can be mapped on a memristive network using a series of memristive devices and resistors. By exploiting the analog computations performed by solving Kirchhoff's Current Laws in a parallel manner [21], memristive networks have been shown to be capable of computing shortest paths in a given maze,

leveraging on the dynamic adjustment of their intrinsic conductance. This computation method has also been extended to show how multiple computations can be performed. Furthermore, this concurrent solution method can also be exploited to include 3D spaces, where shortest paths through stacks of 2D arrays can be efficiently determined by performing a single step via employing distinct voltage sources and ground terminals to the entrances and exits of the network. Such networks, if implemented in hardware, have great application prospects and can be used to solve many optimisation problems in various fields.

ACKNOWLEDGMENTS

The authors wish to acknowledge the financial support of the CHIST-ERA ERANet EPSRC EP/J00801X/1 and EP/K017829/1.

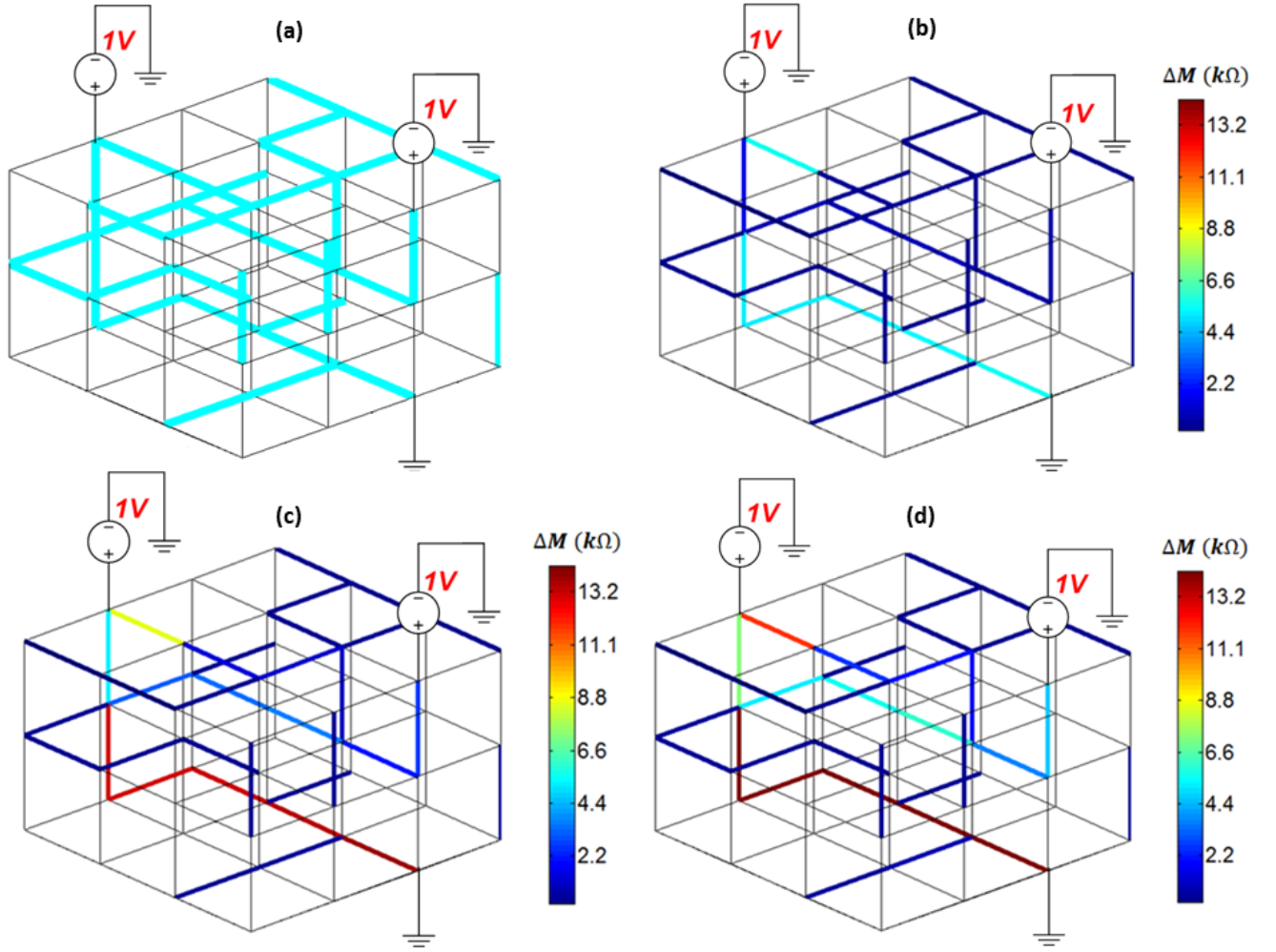


FIG. 11. (Colour Online) Maze shown in a $4 \times 4 \times 3$ Memristive Network. Paths of the maze highlighted in light blue (a), with entrances and exits indicated by 1V and Ground respectively. Solution to the network shown by simulations after 1s (b), 5s (c) and 10s (d).

-
- [1] K. M. Chandy and J. Misra, Commun. ACM **25**, 833 (1982).
 - [2] A. Schrijver, Handbook of Discrete Optimization, 1 (2005).
 - [3] B. Fortz, J. Rexford, and M. Thorup, Communications Magazine, IEEE **40**, 118 (2002).
 - [4] S. Pallottino and M. G. Scutella, Equilibrium and advanced transportation modelling **245**, 281 (1998).
 - [5] J. J. Hopfield and D. W. Tank, Biological Cybernetics **52**, (1985).
 - [6] G. Gallo and S. Pallottino, Netflow at Pisa, 38 (1986).
 - [7] S. Mishra and P. Bande, in *Signal Image Technology and Internet Based Systems, 2008. SITIS '08. IEEE International Conference on* (2008) pp. 86–93.
 - [8] T. Nakagaki, H. Yamada, and A. Toth, Nature (2000).
 - [9] T. Nakagaki, H. Yamada, and M. Hara, Biophysical chemistry **107**, 1 (2004).
 - [10] A. M. Reynolds, Phys. Rev. E **81**, 062901 (2010).
 - [11] I. Lagzi, S. Soh, P. J. Wesson, K. P. Browne, and B. A. Grzybowski, - Journal of the American Chemical Society **132**, 1198 (2010).
 - [12] L. Chua, Circuit Theory, IEEE Transactions on **18**, 507 (1971).
 - [13] L. Chua and S. M. Kang, Proceedings of the IEEE **64**, 209 (1976).
 - [14] T. Prodromakis and C. Toumazou, in *Electronics, Circuits, and Systems (ICECS), 2010 17th IEEE International Conference on* (IEEE, 2010) pp. 934–937.
 - [15] T. Prodromakis, C. Toumazou, and L. Chua, Nat Mater **11**, 478 (2012).
 - [16] Y. V. Pershin and M. Di Ventra, Advances in Physics **60**, 145 (2011).
 - [17] D. B. Strukov, G. S. Snider, D. R. Stewart, and R. S. Williams, Nature **453**, 80 (2008).
 - [18] H. Kim, M. P. Sah, C. Yang, and L. O. Chua, in *12th International Workshop on Cellular Nanoscale Networks*

- and Their Applications (CNNA)*, 2010 (2010) pp. 1–6.
- [19] S. H. Jo, T. Chang, I. Ebong, B. B. Bhadviya, P. Mazumder, and W. Lu, *Nano Letters* **10**, 1297 (2010).
 - [20] G. Snider, *IOPScience Nanotechnology* **8**, 365202 (2007).
 - [21] Y. V. Pershin and M. Di Ventra, *Physical Review E* **84**, 046703 (2011).
 - [22] R. Williams, *Spectrum*, *IEEE* **45**, 28 (2008).
 - [23] Y. N. Joglekar and J. Stephen, *European Journal of Physics* **30**, 661 (2009).
 - [24] F. Jiang and B. E. Shi, in *Circuit Theory and Design, 2009. ECCTD 2009. European Conference on* (IEEE, 2009) pp. 181–184.
 - [25] A. Gelencser, T. Prodromakis, C. Toumazou, and T. Roska, *Physical Review E* **85**, 041918 (2012).
 - [26] Z. Biolek, D. Biolek, and V. Biolkova, *Radioengineering* **18**, 210 (2009).
 - [27] T. Prodromakis, B. P. Peh, C. Papavassiliou, and C. Toumazou, *Electron Devices, IEEE Transactions on* **58**, 3099 (2011).
 - [28] D. R. Reyes, M. M. Ghanem, G. M. Whitesides, and A. Manz, *Lab on a Chip* **2**, 113 (2002).
 - [29] E. W. Dijkstra, *Numerische mathematik* **1**, 269 (1959).
 - [30] M. Sharma and K. Robeonics, in *Future Computer and Communication, 2009. ICFCC 2009. International Conference on* (2009) pp. 581–585.
 - [31] R. W. Floyd, *Communications of the ACM* **5**, 345 (1962).
 - [32] T. H. Cormen, C. E. Leiserson, and R. L. Rivest, *Introduction to Algorithms*, 1st ed. (MIT Press and McGraw-Hill, 1990).
 - [33] D. Coppersmith and S. Winograd, *Journal of Symbolic Computation* **9**, 251 (1990).